# *Dietly* *Üst Mimari Tasarım Raporu*

*Yazan ve Düzenleyenler:*

- *Yusuf Talha Yılmaz 434385*
- *Cem Onat Satır 434395*
- *Muhammed Ali Rıza Bağcı 434403*

## *Projenin Amacı ve Tanımı:*

***Dietly**, kullanıcıların kişisel sağlık hedeflerine uygun bir şekilde beslenme düzeni oluşturmalarını, günlük kalori takibi yapmalarını ve sağlıklı yaşam alışkanlıkları geliştirmelerini amaç edinen bir mobil uygulamadır. Uygulama, kullanıcı dostu arayüzü ve minimal tasarımı ile kullanıcıların rahatlıkla kullanabileceği şekilde tasarlanmıştır.*

### *Genel Hedefleri:*

- *Kullanıcıların günlük kalori takibini kolaylaştırmak.*
- *Kişiselleştirilmiş diyet programları oluşturmalarını sağlamak.*
- *Basit, kullanışlı ve şık bir arayüzle kullanıcı deneyimini iyileştirmek.*
- ***Firebase** altyapısı ile güvenli bir kimlik doğrulama ve veri yönetimi sağlamak.*

## *Genel Sistem Mimarisi:*

***Dietly** mobil uygulaması, katmanlı mimari desenini takip ederek geliştirilecektir. Bu yaklaşım, uygulamanın farklı işlevsel bölümlerini birbirinden ayırarak daha modüler, bakımı kolay ve ölçeklenebilir bir yapı sunar.*

## *Sistemler ve Alt Sistemler:*

*Uygulamanın fonksiyonel gereksinimleri doğrultusunda(bkz. GereksinimAnalizRaporu) farklı alt sistemlere bölünmesi, geliştirme sürecinde modülerliği ve esnekliği artırmaktadır. Projenin şu anki mimarisinde aşağıdaki alt sistemler yer almaktadır:*

### *1) Kullanıcı Yönetim Modülü:*

- ***Firebase** Authentication kullanılarak kullanıcıların kayıt olmasını ve giriş yapmasını sağlar.*
- *E-posta ve şifre kontrolü, hata mesajları ve yönlendirme işlemlerini içerir.*

### *2) Splash ve Onboarding Modülü:*

- *Uygulama açıldığında kısa süreli bir animasyon gösterilerek kullanıcıya ilk izlenim sağlanır.*
- *Onboarding ekranları ile uygulama özellikleri tanıtılır.(İleri sürümlerde...)*

### *3) Profil Yönetimi Modülü:*

- *Kullanıcıların kişisel bilgilerini girebileceği ve güncelleyebileceği ekranları içerir.*
- *Bu veriler kullanıcıya özel öneriler sunulmasında kullanılır.*

### *4) UI Bileşenleri:*

***Dietly** uygulamasının kullanıcı arayüzü, Flutter'ın sunduğu modern widget altyapısı kullanılarak sade, şık ve kullanıcı dostu bir yapıda tasarlanmıştır. Uygulamanın modülerliği göz önünde bulundurularak yeniden kullanılabilir widget yapıları oluşturulmuştur. Bu sayede hem kod tekrarının önüne geçilmiş hem de bakım kolaylığı sağlanmıştır.*

#### ***4.1) Splash Screen Bileşenleri:***

- ***Lottie Animasyonu(Lottie.asset)***

*Açılışta gösterilen animasyon, kullanıcıya uygulamanın yüklenmekte olduğunu göstererek profesyonel bir izlenim sağlar. "assets/animations/splash_screen.json" dosyasından okunur ve **Dietly** yazısıyla birlikte marka algısını güçlendirmek ve kullanıcıyı karşılamayı amaç edinmektedir.*

### *4.2) Auth(Login ve Register) Screen Bileşenleri:*

- ***CustomTextField Widget'ı***

*Kullanıcıdan e-posta ve şifre bilgisi almak için kullanılan özelleştirilmiş bir "TextField" bileşeni. İçerisinde bulunan Placeholder (hintText), prefixIcon, obscureText; gibi parametrelerle esnek genişletilebilir ve reusable (yeniden kullanılabilir) yapı.*

- ***PrimaryButton Widget'ı***

*"Login", "Register" gibi işlemleri başlatmak için kullanılan temel buton bileşeni. Buton metni, renk, tıklama işlevi (onPressed) gibi parametreler ile kişiselleştirilebilir. Sabit stil, kenar yuvarlaklığı ve gölge efekti ile tutarlı tasarım sağlar.*

### *4.3) Reusable UI Component Yaklaşımı:*

*Uygulamada birçok bileşen, farklı ekranlarda tekrar kullanılabilir hale getirilmiştir. Bu modülerlik sayesinde:*

- *Her bileşen gerektiğinde sadece bir dosyada güncellenerek tüm uygulamada değiştirilebilir.*
- *Geliştirme süreci hızlanır ve UI tutarlılığı sağlanır.*

### *4.4) Temalar ve Renk Paleti:*

- *Uygulamanın ana renk paleti(bkz. ColorPalette), markalaşma açısından istikrar sağlar.*
- *Arka plan, butonlar ve metinlerde sade ve kontrast sağlayan renkler tercih edilmiştir.*
- *Google Fonts kullanılarak metinlerde profesyonel bir görünüm amaçlanmıştır.*

### *4.5) Responsive (Duyarlı) Tasarım:*

*Tüm bileşenler, cihaz boyutuna göre esneklik gösterecek şekilde "MediaQuery" veya "Expanded", "Flexible" gibi yapılarla tasarlanıp farklı ekran boyutlarında (telefon, tablet) sorunsuz çalışabilmesi sağlanmıştır.*

---

# *Katmanlı Mimari:*

***Dietly** uygulaması, katmanlı mimari prensiplerine uygun olarak geliştirilmektedir. Bu yapı sayesinde uygulamanın sürdürülebilirliği ve test edilebilirliği artırılmıştır. Temel olarak üç katmandan oluşur:*

- ***Sunum Katmanı (Presentation Layer):** Kullanıcı arayüzü bileşenlerini içerir. Kullanıcı ile uygulama arasındaki etkileşimi yönetir.*
- ***İş Mantığı Katmanı (Business Logic Layer):** Uygulamanın kurallarını, servis mantığını ve verilerin işlenmesini sağlar.*
- ***Veri Katmanı (Data Layer):** Firebase Firestore ile veri kayıt, okuma ve güncelleme işlemlerini yönetir.*

---

# *Temel Sınıflar:*

## *1) Sunum Katmanı Sınıfları*

- ***SplashScreen:** Uygulamanın açılışında animasyon gösteren ekran*
- ***LoginScreen:** Kullanıcı giriş ekranı*
- ***RegisterScreen:** Yeni kullanıcı kaydı ekranı*
- ***ProfileScreen:** Kullanıcı profilini görüntüleme ve düzenleme ekranı*
- ***FoodSearchScreen:** Besin arama ekranı*
- ***FoodLogScreen:** Besin günlüğü ekranı*
- ***RecipeListScreen:** Tarif arama ve listeleme ekranı*
- ***ProgressScreen:** Günlük veya haftalık ilerleme ekranı*

## *2) İş Mantığı Katmanı Sınıfları*

- ***AuthService:** Firebase Authentication ile giriş/kayıt işlemlerini yönetir*
- ***FoodService:** Besin verileri işlemlerini yürütür, kalori hesaplamalarını yapar*
- ***RecipeService:** Tarif önerilerini sağlar*
- ***ProgressService:** Kullanıcının ilerleme takibini ve analizini yapar*

## *3) Veri Katmanı Sınıfları*

- ***UserRepository**: Kullanıcı verilerini Firestore'a yazar/okur*
- ***FoodRepository**: Besin verileri için CRUD işlemleri*
- ***RecipeRepository:** Tarif verilerini yönetir*

---

# *Rutin Akışlar:*

***Kullanıcı Kayıt Rutini:***

1. *RegisterScreen'de bilgiler girilir*
2. *AuthService.createUser() çağrılır*
3. *Firestore'a veri UserRepository ile kaydedilir*
4. *Kullanıcı ProfileScreen’e yönlendirilir*

***Giriş Rutini:***

1. *Kullanıcı LoginScreen'de bilgiler girer*
2. *AuthService.signIn() çağrılır*
3. *Kullanıcı FoodLogScreen’e aktarılır*

***Besin Ekleme Rutini:***

1. *Kullanıcı FoodSearchScreen'de arama yapar*
2. *FoodService.searchFood() ile sonuçlar getirilir*
3. *Kullanıcı log ekranından besin ekler*
4. *addFoodToLog() ile veri Firestore'a gönderilir*

***Tarif Arama Rutini:***

1. *RecipeListScreen'de arama yapılır*
2. *RecipeService.searchRecipes() çağrılır ve liste görüntülenir*

***İlerleme Takibi Rutini:***

1. *Kullanıcı ProgressScreen'e girer*
2. *ProgressService.getDailyProgress() çağrılır*
3. *Veriler grafiksel olarak sunulur*

---

# *Kullanılan Teknolojiler:*

| ***Katman*** | ***Kullanılan Teknoloji/Kütüphane*** | ***Açıklama*** |
|---|---|---|
| *Sunum Katmanı* | *Flutter, Dart, Google Fonts, Lottie* | *Modern ve animasyon destekli kullanıcı arayüzü için Flutter kullanılmıştır.* |
| *İş Mantığı Katmanı* | *Provider (ya da Riverpod gibi durum yönetim kütüphaneleri)* | *UI ile iş mantığı arasında veri paylaşımı ve kontrolü sağlar.* |
| *Veri Katmanı* | *Firebase Authentication, Cloud Firestore* | *Kullanıcı yönetimi ve veri saklama işlemleri için güvenli bir yapı sunar.* |
| *Diğer* | *Git, GitHub, Firebase CLI, Android Studio* | *Sürüm kontrolü, geliştirme ortamı ve yapılandırma araçları.* |

---

# *Kodlama Standartları:*

- ***Dart Stil Rehberi:*** *Dokümantasyona uygun bir şekilde yazılır paketler resmi siteden(bkz.PubDev) alınır*
- ***Camel Case:*** *Değişken ve fonksiyon isimlendirme; Fonksiyonlar küçük harfle başlar ama CamelCase ile devam eder: backgroundColor()*
- ***Yorum Satırları:*** *Karmaşık bloklar açıklanır(İleri versiyonlarda kaldırılabilir.)*
- ***Hata Yönetimi:*** *try-catch ile anlamlı geri bildirim Örnek olarak:*

---

  - *try {*
    *await Auth().createUser(email: email, password: password);*
    *registrationSuccessful = true;*
    *} catch (e) {*
    *errorMessage = e.toString();*
    *}*

---

- ***Tutarlı UI:*** *Renk, font, buton ve form alanlarında standart*
- ***State Yönetimi:*** *Sayfalar arası veri akışı sağlanır ve ekranları güncel tutar (createUser).*
- ***Klasör Yapısı:*** *Katmanlı, modüler ayrışır:*

*lib/ *Lib kodların tutulduğu klasör**

*├── screens/ *Uygulamada kullanılan ekranlar**

*│ ├── login_screen.dart *Giriş ekranı**

*│ └── register_screen.dart *Kayıt Ekranı**

*│ └── splash_screen.dart *Yükleme ekranı**

*│ └── profile_creation_screen.dart *Profil oluşturma ekranı**

*├── services/ *Firebase servisleri**

*│ └── auth.dart *Auth(Giriş,Kayıt) işlemlerinin veritabanında tutulması**

*├── widgets/ *Tasarımda kullanılan widget yapıları**

*│ └── bottom_shape.dart *Uygulamamızın arayüzünün alt kısmında bulunan 2 üçgen 1 beşgen**

*└── main.dart *Uygulamamızın ana dosyası**

*└── firebase_options.dart *Firebase Confugration**

---

# *Test Yaklaşımları:*

*Kaliteli ve güvenilir bir uygulama için aşağıdaki test türleri uygulanacaktır:*

- ***Unit Testler:*** *Servis ve yardımcı sınıfların doğruluğunu test eder*
- ***Widget Testler:*** *UI bileşenlerinin beklendiği gibi çalışıp çalışmadığını kontrol eder*
- ***Entegrasyon Testleri:*** *Katmanlar arası uyumu test eder*
- ***UI Testleri:*** *Manuel ve otomatik olarak, uygulama arayüzünün işlevselliği test edilir*
- ***Firebase Emülatörü Kullanımı:*** *Firebase servislerinin yerel ortamda test edilmesini sağlar*

---

# *Sonuç:*

*Bu mimari tasarım raporu, **Dietly** uygulamasının profesyonel ve sürdürülebilir şekilde geliştirilmesine yön vermek amacıyla hazırlanmıştır. Katmanlı yapı, modern araçlar, açık kodlama standartları ve güçlü test yaklaşımları ile birlikte **Dietly**'nin uzun vadede gelişmeye açık, kullanıcı dostu ve sağlıklı yaşamı destekleyen bir uygulama olması hedeflenmektedir.*

*Not: Uygulamayı geliştirmeye devam edeceğiz...*